# AN EVALUATION OF LEXICALIZATION IN PARSING

Aravind K. Joshi and Yves Schabes
Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA 19104-6389
joshi/schabes@linc.cis.upenn.edu

### Abstract

In this paper, we evaluate a two-pass parsing strategy proposed for the so-called 'lexicalized' grammar. In 'lexicalized' grammars (Schabes, Abeillé and Joshi, 1988), each elementary structure is systematically associated with a lexical item called *anchor*. These structures specify extended domains of locality (as compared to CFGs) over which constraints can be stated. The 'grammar' consists of a lexicon where each lexical item is associated with a finite number of structures for which that item is the anchor. There are no separate grammar rules. There are, of course, 'rules' which tell us how these structures are combined.

A general two-pass parsing strategy for 'lexicalized' grammars follows naturally. In the first stage, the parser selects a set of elementary structures associated with the lexical items in the input sentence, and in the second stage the sentence is parsed with respect to this set. We evaluate this strategy with respect to two characteristics. First, the amount of filtering on the entire grammar is evaluated: once the first pass is performed, the parser uses only a subset of the grammar. Second, we evaluate the use of non-local information: the structures selected during the first pass encode the morphological value (and therefore the position in the string) of their anchor; this enables the parser to use non-local information to guide its search.

We take Lexicalized Tree Adjoining Grammars as an instance of lexicalized grammar. We illustrate the organization of the grammar. Then we show how a general Earley-type TAG parser (Schabes and Joshi, 1988) can take advantage of lexicalization. Empirical data show that the filtering of the grammar and the non-local information provided by the two-pass strategy improve the performance of the parser.

# 1 LEXICALIZED GRAMMARS

Most current linguistic theories give lexical accounts of several phenomena that used to be considered purely syntactic. The information put in the lexicon is thereby increased in both amount and complexity: see, for example, lexical rules in LFG (Kaplan and Bresnan, 1983), GPSG (Gazdar, Klein, Pullum and Sag, 1985), HPSG (Pollard and Sag, 1987), Combinatory Categorial Grammars (Steedman 1985, 1988), Karttunen's version of Categorial Grammar (Karttunen 1986, 1988), some versions of GB theory (Chomsky 1981), and Lexicon-Grammars (Gross 1984).

We say that a grammar is 'lexicalized' if it consists of:[1]

- a finite set of structures each associated with a lexical item; each lexical item will be called the *anchor* of the corresponding structure; the structures define the domain of locality over which constraints are specified; constraints are local with respect to their anchor;
- an operation or operations for composing the structures.

Notice that Categorial Grammars (as used for example by Ades and Steedman, 1982 and Steedman, 1985 and 1988) are 'lexicalized' according to our definition since each basic category has a lexical item associated with it.

A general two-step parsing strategy for 'lexicalized' grammars follows naturally. In the first stage, the parser selects a set of elementary structures associated with the lexical items in the input sentence, and in the second stage the sentence is parsed with respect to this set. The strategy is independent of the nature of the elementary structures in the underlying grammar. In principle, any parsing algorithm can be used in the second stage.

[1] By 'lexicalization' we mean that in each structure there is a lexical item that is realized. We do not mean simply adding feature structures (such as head) and unification equations to the rules of the formalism.

The first step selects a relevant subset of the entire grammar, since only the structures associated with the words in the input string are selected for the parser. In the worst case, this filtering would select the entire grammar. The number of structures filtered during this pass depends on the nature of the input string and on characteristics of the grammar such as the number of structures, the number of lexical entries, the degree of lexical ambiguity, and the languages it defines.

Since the structures selected during the first step encode the morphological value of their anchor (and therefore its position in the input string), the first step also enables the parser to use non-local information to guide its search. The encoding of the value of the anchor of each structure constrains the way the structures can be combined. It seems that this information is particularly useful for parsing algorithms that have some top-down behavior.

This parsing strategy is general and any standard parsing technique can be used in the second step. Perhaps the advantages of the first step could be captured by some other technique. However this strategy is extremely simple and is consistent with the linguistic motivations for lexicalization.

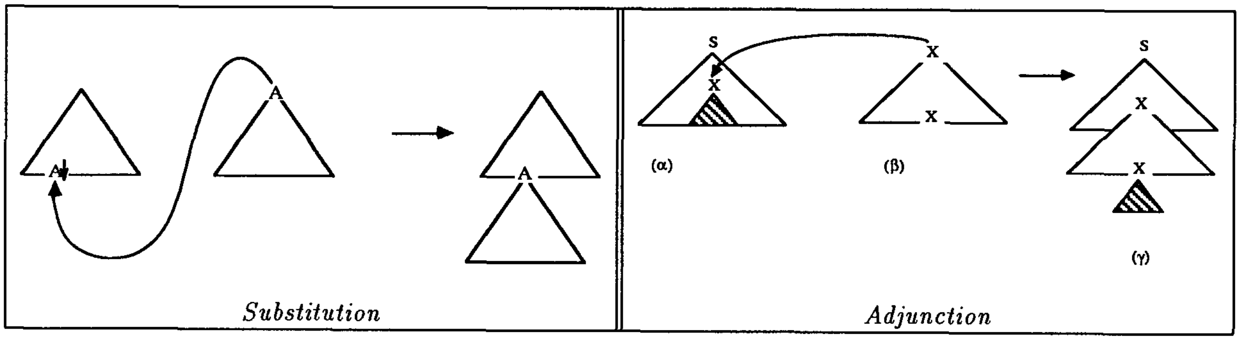
## 2 LEXICALIZED TAGS

Not every grammar is in a 'lexicalized' form.[2] In the process of lexicalizing a grammar, we require that the 'lexicalized' grammar produce not only the same language as the original grammar, but also the same structures (or tree set).

For example, a CFG, in general, will not be in a 'lexicalized' form. The domain of locality of CFGs can be easily extended by using a tree rewriting grammar (Schabes, Abeillé and Joshi, 1988) that uses only substitution as a combining operation. This tree rewriting grammar consists of a set of trees that are not restricted to be of depth one (as in CFGs). Substitution can take place only on non-terminal nodes of the frontier of each tree. Substitution replaces a node marked for substitution by a tree rooted by the same label as the node (see Figure 1; the substitution node is marked by a down arrow ↓).

However, in the general case, CFGs cannot be 'lexicalized', if only substitution is used. Furthermore, in general, there is not enough freedom to choose the anchor of each structure. This is important because we want the choice of the anchor for a given structure to be determined on purely linguistic grounds.

If adjunction is used as an additional operation to combine these structures, CFGs can be lexicalized. Adjunction builds a new tree from an auxiliary tree $\beta$ and a tree $\alpha$ . It inserts an auxiliary tree in another tree (see Figure 1). Adjunction is more powerful than substitution. It can weakly simulate substitution, but it also generates languages that could not be generated with substitution.[3]



*Substitution*

*Adjunction*

Figure 1: *Combining operations*

Substitution and adjunction enable us to lexicalize CFGs. The 'anchors' can be freely chosen (Schabes, Abeillé and Joshi, 1988). The resulting system now falls in the class of mildly context-sensitive languages

[2] Notice the similarity of the definition of 'lexicalized' grammar with the offline parsibility constraint (Kaplan and Bresnan 1983). As consequences of our definition, each structure has at least one lexical item (its anchor) attached to it and all sentences are finitely ambiguous.

[3] It is also possible to encode a context-free grammar with auxiliary trees using adjunction only. However, although the languages correspond, the set of trees do not correspond.

(Joshi, 1985). Elementary structures of extended domain of locality combined with substitution and adjunction yield Lexicalized TAGs.

TAGs were first introduced by Joshi, Levy and Takahashi (1975) and Joshi (1985). For more details on the original definition of TAGs, we refer the reader to Joshi (1985), Kroch and Joshi (1985), or Vijay-Shanker (1987). It is known that Tree Adjoining Languages (TALs) are mildly context sensitive. TALs properly contain context-free languages.

TAGs with substitution and adjunction are naturally lexicalized.[4] A Lexicalized Tree Adjoining Grammar is a tree-based system that consists of two finite sets of trees: a set of initial trees, $I$ and a set of auxiliary trees $A$ (see Figure 2). The trees in $I \cup A$ are called **elementary trees**. Each elementary tree is constrained to have at least one terminal symbol which acts as its anchor.

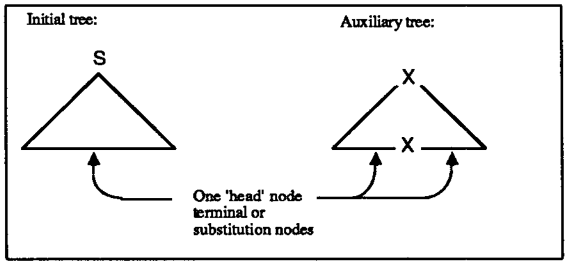


Figure 2: *Schematic initial and auxiliary trees*

The **tree set** of a TAG $G$, $\mathcal{T}(G)$ is defined to be the set of all derived trees starting from S-type initial trees in $I$. The **string language** generated by a TAG, $\mathcal{L}(G)$, is defined to be the set of all terminal strings of the trees in $\mathcal{T}(G)$.

By lexicalizing TAGs, we have associated lexical information to the 'production' system encoded by the TAG trees. We have therefore kept the computational advantages of 'production-like' formalisms (such as CFGs, TAGs) while allowing the possibility of linking them to lexical information. Formal properties of TAGs hold for Lexicalized TAGs.

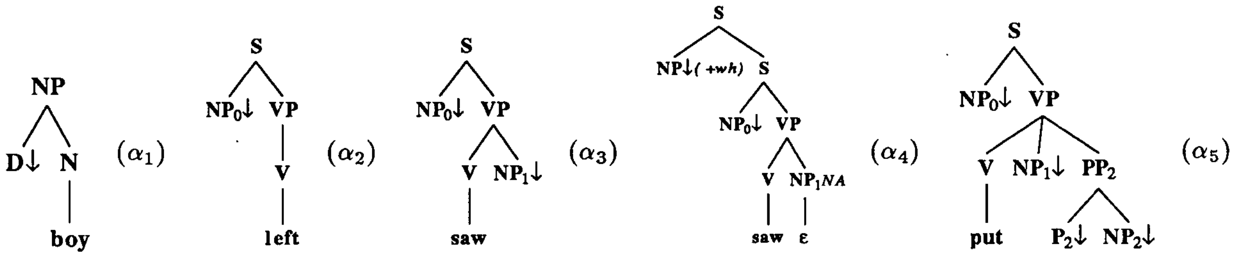
As first shown by Kroch and Joshi (1985), the properties of TAGs permit us to encapsulate diverse syntactic phenomena in a very natural way. TAG's extended domain of locality and its factoring recursion from local dependencies lead, among other things, to localizing the so-called unbounded dependencies. Abeillé (1988a) uses the distinction between substitution and adjunction to capture the different extraction properties between sentential subjects and complements. Abeillé (1988c) makes use of the extended domain of locality and lexicalization to account for NP island constraint violations in light verb constructions; in such cases, extraction out of NP is to be expected, without the use of reanalysis. The relevance of Lexicalized TAGs to idioms has been suggested by Abeillé and Schabes (1989).

We will now give some examples of structures that appear in a Lexicalized TAG lexicon.

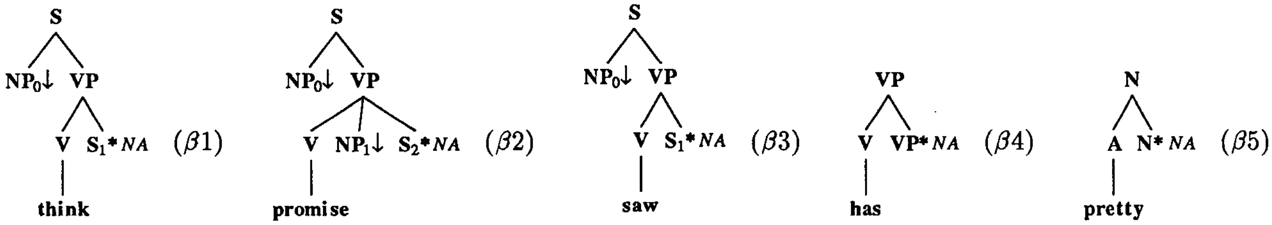
Some examples of initial trees are (for simplicity, we have omitted unification equations associated with the trees):[5]

[4] In some earlier work of Joshi (1969, 1973), the use of the two operations 'adjoining' and 'replacement' (a restricted case of substitution) was investigated both mathematically and linguistically. However, these investigations dealt with string rewriting systems and not tree rewriting systems.

[5] The trees are simplified and the feature structures on the trees are not displayed. $\downarrow$ is the mark for substitution nodes, $*$ is the mark for the foot node of an auxiliary tree and $NA$ stands for null adjunction constraint. This is the only adjunction constraint not indirectly stated by feature structures. We put indices on some non-terminals to express syntactic roles (0 for subject, 1 for first object, etc.). The index shown on the empty string ($\epsilon$) and the corresponding filler in the same tree is for the purpose of indicating the filler-gap dependency.



Examples of auxiliary trees (they correspond to predicates taking sentential complements or modifiers):



In this approach, the argument structure is not just a list of arguments. It is the syntactic structure constructed with the lexical value of the predicate and with all the nodes of its arguments that eliminates the redundancy often noted between phrase structure rules and subcategorization frames.[6]

## 2.1 ORGANIZATION OF THE GRAMMAR

A Lexicalized TAG is organized into two major parts: **a lexicon** and **tree families**, which are sets of trees.[7] TAG's factoring recursion from dependencies, the extended domain of locality of TAGs, and lexicalization of elementary trees make Lexicalized TAG an interesting framework for grammar writing. Abeillé (1988b) discusses the writing of a Lexicalized TAG for French. Abeillé, Bishop, Cote and Schabes (1989) similarly discuss the writing of a Lexicalized TAG grammar for English.

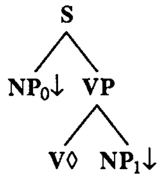
### 2.1.1 TREE FAMILIES

A **tree family** is essentially a set of sentential trees sharing the same argument structure abstracted from the lexical instantiation of the anchor (verb, predicative noun or adjective). Because of the extended domain of locality of Lexicalized TAG, the argument structure is not stated by a special mechanism but is implicitly stated in the topology of the trees in a tree family. Each tree in a family can be thought of as all possible syntactic 'transformations' of a given argument structure. Information (in the form of feature structures) that is valid independent of the value of the anchor is stated on the tree of the tree family. For example, the agreement between the subject and the main verb or auxiliary verb is stated on each tree of the tree family. Currently, the trees in a family are explicitly enumerated.

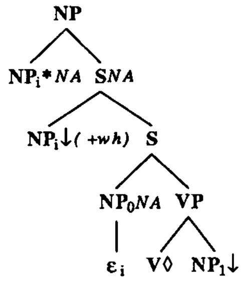
[6] Optional arguments are stated in the structure.

[7] There are actually two types of lexicons: a morphological lexicon which lists the possible morphological variations for a word and a syntactic lexicon which associates the variations of a given word to syntactic elementary trees. In this paper we will ignore the morphological lexicon and we will use the term lexicon for refering to the syntactic lexicon.

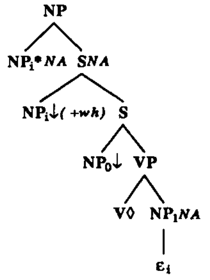
The following trees, among others, compose the tree family of verbs taking one object (the family is named *np0Vnp1*):[8]



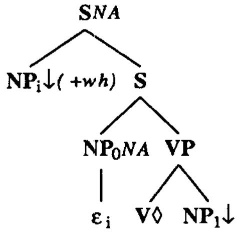
(*αnp0Vnp1*)



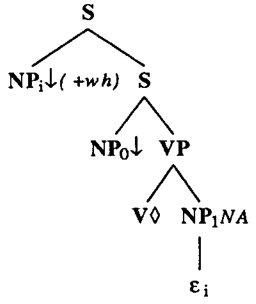
(*βR0np0Vnp1*)



(*βR1np0Vnp1*)



(*αW0np0Vnp1*)



(*αW1np0Vnp1*)

*αnp0Vnp1* is an initial tree corresponding to the declarative sentence, *βR0np0Vnp1* is an auxiliary tree corresponding to a relative clause where the subject has been relativized, *βR1np0Vnp1* corresponds to the relative clause where the object has been relativized, *αW0np0Vnp1* is an initial tree corresponding to a wh-question on the subject, *αW1np0Vnp1* corresponds to a wh-question on the object.

### 2.1.2 THE LEXICON

The **lexicon** is the heart of the grammar. It associates a word with tree families or trees. Words are not associated with basic categories as in a CFG-based grammar, but with tree-structures corresponding to minimal linguistic structures. Multi-level dependencies can thus be stated in the lexicon.

It also states some word-specific feature structure equations (such as the agreement value of a given verb) that have to be added to the ones already stated on the trees (such as the equality of the value of the subject and verb agreements).

An example of a lexical entry follows:

```
loves, V {V.b:<mode>=ind,
          V.b:<agr pers>= 3,
          V.b:<agr num>= singular,
          V.b:<tense>=present} :np0Vnp1.
```

It should be emphasized that in our approach the category of a word is not a non-terminal symbol but a multi-level structure corresponding to minimal linguistic structures: sentences (for predicative verbs, nouns and adjectives) or phrases (NP for nouns, AP for adjectives, PP for prepositions yielding adverbial phrases).

## 2.2 PARSING LEXICALIZED TAGs

An Earley-type parser for TAGs has been developed by Schabes and Joshi (1988). It is a general TAG parser. It handles adjunction and substitution. It can take advantage of lexicalization. It uses the structures selected after the first pass to parse the sentence. The parser is able to use the non-local information given by the first step to filter out prediction and completion states.

### 2.2.1 TAKING ADVANTAGE OF LEXICALIZATION

If an offline behavior is adopted, the Earley-type parser for TAGs can be used with no modification for parsing Lexicalized TAGs. First the trees corresponding to the input string are selected and then the parser parses the input string with respect to this set of trees.

However, Lexicalized TAGs simplify some cases of the algorithm. For example, since by definition each tree has at least one lexical item attached to it (its anchor), it will not be the case that a tree can be predicted

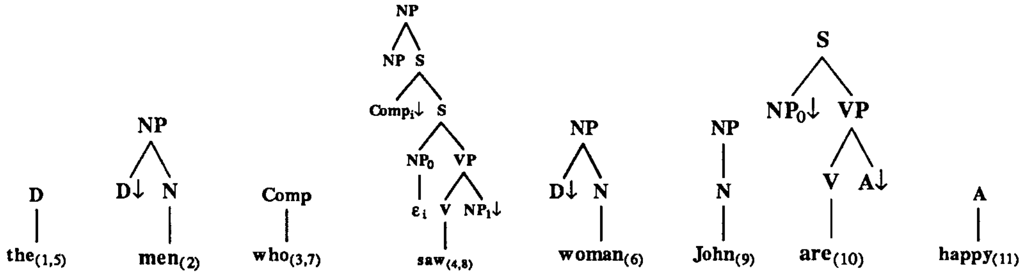
[8] The trees are simplified. ◇ is the mark for the node under which the lexical insertion of the anchor is performed.

for substitution and completed in the same states set. Similarly, it will not be the case that an auxiliary tree can be left predicted for adjunction and right completed in the same states set.

But most importantly the algorithm can be extended to take advantage of Lexicalized TAGs. Once the first pass has been performed, a subset of the grammar is selected. Each structure encodes the morphological value (and therefore the positions in the string) of its anchor. Identical structures with different anchor values are merged together (by identical structures we mean identical trees and identical information, such as feature structures, stated on those trees).[9] This enables us to use the anchor position information while processing efficiently the structures. For example, given the sentence

The $_1$ men $_2$ who $_3$ saw $_4$ the $_5$ woman $_6$ who $_7$ saw $_8$ John $_9$ are $_{10}$ happy $_{11}$

the following trees (among others) are selected after the first pass:[10]



The trees for men and for woman are distinguished since they carry different agreement feature structures (not shown in the figure).

Notice that there is only one tree for the relative clauses introduced by saw but that its anchor position can be 4 or 8. Similarly for who and the.

The anchor positions of each structure impose constraints on the way that the structures can be combined (the anchor positions must appear in increasing order in the combined structure). This helps the parser to filter out predictions or completions for adjunction or substitution. For example, the tree corresponding to men will not be predicted for substitution in any of the trees corresponding to saw since the anchor positions would not be in the right order.

We have been evaluating the influence of the filtering of the grammar and the anchor position information on the behavior of the Earley-type parser. We have conducted experiments on a feature structure-based Lexicalized English TAG whose lexicon defines 200 entries associated with 130 different elementary trees (the trees are differentiated by their topology and their feature structures but not by their anchor value). Twenty five sentences of length ranging from 3 to 14 words were used to evaluate the parsing strategy. For each experiment, the number of trees given to the parser and the number of states were recorded.

In the first experiment (referred to as *one pass*, *OP*), no first pass was performed. The entire grammar (i.e., the 130 trees) was used to parse each sentence. In the second experiment (referred to as *two passes no anchor*, *NA*), the two-pass strategy was used but the anchor positions were not used in the parser. And in the third experiment (referred to as *two passes with anchor*, *A*), the two-pass strategy was used and the information given by the anchor positions was used by the parser.

The average behavior of the parser for each experiment is given in Figure 3. The first pass filtered on average 85% (always at least 75%) of the trees. The filtering of the grammar by itself decreased by 86% the number of states $((NA - OP)/OP)$. The additional use of the information given by the anchor positions further decreased by 50% $((A - NA)/NA)$ the number of states. The decrease given by the filtering of the grammar and by the information of the anchor positions is even bigger on the number of attempts to add a state (not reported in the table).[11]

This set of experiments shows that the two-pass strategy increases the performance of the Earley-type parser for TAGs. The filtering of the grammar affects the parser the most. The information given by anchor

[9] Unlike our previous suggestions (Schabes, Abeillé and Joshi, 1988), we do not distinguish each structure by its anchor position since it increases unnecessarily the number of states of the Earley parser. By factoring recursion, the Earley parser enables us to process only once parts of a tree that are associated with several lexical items selecting the same tree. However, if termination is required for a pure top-down parser, it is necessary to distinguish each structure by its anchor position.

[10] The example is simplified to illustrate our point.

[11] A state is effectively added to a states set if it does not exist in the set already.

position in the first pass allows further improvement of the parser's performance (- 50% of the number of states on the set of experiments). The bottom-up non-local information given by the anchor positions improves the top-down component of the Earley-type parser.

| | (NA-OP)/OP (%) | (A-OP)/OP (%) | (A - NA)/NA (%) |
|---|---|---|---|
| # trees | -85 | -85 | 0 |
| # states | -86 | -93 | -50 |

Figure 3: *Empirical evaluation of the two-pass strategy*

We performed our evaluation on a relatively small grammar and we did not evaluate the variations across grammars. The lexical degree of ambiguity of each word, the number of structures in the grammar, the number of lexical entries, and the length (and nature) of the input sentences are parameters to be considered. Although it might appear easy to conjecture the influence of these parameters, the actual experiments are difficult to perform since statistical data on these parameters are hard to obtain. We hope to perform some limited experiments along those lines.

# 3 CONCLUSION

In 'lexicalized' grammars, each elementary structure is systematically associated with a lexical anchor. These structures specify extended domains of locality (as compared to the domain of locality in CFGs) over which constraints can be stated. The 'grammar' consists of a lexicon in which each lexical item is associated with a finite number of structures for which that item is the anchor.

Lexicalized grammars suggest a natural two-step parsing strategy. The first step selects the set of structures corresponding to each word in the sentence. The second step tries to combine the selected structures.

We take Lexicalized TAGs as an instance of lexicalized grammar. We illustrate the organization of the grammar Then we show how the Earley-type parser can take advantage of the two-step parsing strategy. Experimental data show that its performance is thereby drastically improved. The first pass not only filters the grammar used by the parser to produce a relevant subset but also enables the parser to use non-local bottom-up information to guide its search. In Schabes and Joshi (1989) it is also shown that Lexicalization guarantees termination of the parsing algorithm of feature structures for Lexicalized TAGs without a special mechanism such as the use of restrictors.

The organization of lexicalized grammars, the simplicity and effectiveness of the two-pass strategy (some other technique would perhaps achieve similar results) seem attractive from a linguistic point of view and for processing.

# References

Abeillé, Anne, August 1988 (a). Parsing French with Tree Adjoining Grammar: some Linguistic Accounts. In *Proceedings of the* $12^{th}$ *International Conference on Computational Linguistics (COLING'88).* Budapest.

Abeillé, Anne, 1988 (b). *A Lexicalized Tree Adjoining Grammar for French: the General Framework.* Technical Report MS-CIS-88-64, Department of Computer and Information Science, University of Pennsylvania.

Abeillé, Anne, 1988 (c). Light Verb Constructions and Extraction out of NP in Tree Adjoining Grammar. In *Papers from the 24th Regional Meeting of the Chicago Linguistic Society.* Chicago.

Abeillé, Anne and Schabes, Yves, 1989. Parsing Idioms in Tree Adjoining Grammars. In *Fourth Conference of the European Chapter of the Association for Computational Linguistics (EACL'89).* Manchester.

Abeillé, Anne; M., Bishop Kathleen; Cote, Sharon; and Schabes, Yves, 1989. *A Lexicalized Tree Adjoining Grammar for English.* Technical Report, Department of Computer and Information Science, University of Pennsylvania.

Ades, A. E. and Steedman, M. J., 1982. On the Order of Words. *Linguistics and Philosophy* 3:517–558.

Chomsky, N., 1981. *Lectures on Government and Binding.* Foris, Dordrecht.

Gazdar, G.; Klein, E.; Pullum, G. K.; and Sag, I. A., 1985. *Generalized Phrase Structure Grammars.* Blackwell Publishing, Oxford. Also published by Harvard University Press, Cambridge, MA.

Gross, Maurice, 2-6 July 1984. Lexicon-Grammar and the Syntactic Analysis of French. In *Proceedings of the* $10^{th}$ *International Conference on Computational Linguistics (COLING'84).* Stanford.

Joshi, Aravind K., August 1969. Properties of Formal Grammars with Mixed Type of Rules and their Linguistic Relevance. In *Proceedings of the International Conference on Computational Linguistics.* Sanga Saby.

Joshi, Aravind K., 1973. A Class of Transformational Grammars. In M. Gross, M. Halle and Schutzenberger, M.P. (editors), *The Formal Analysis of Natural Languages.* Mouton, La Hague.

Joshi, Aravind K., 1985. How Much Context-Sensitivity is Necessary for Characterizing Structural Descriptions—Tree Adjoining Grammars. In Dowty, D.; Karttunen, L.; and Zwicky, A. (editors), *Natural Language Processing—Theoretical, Computational and Psychological Perspectives.* Cambridge University Press, New York. Originally presented in a Workshop on Natural Language Parsing at Ohio State University, Columbus, Ohio, May 1983.

Joshi, A. K.; Levy, L. S.; and Takahashi, M., 1975. Tree Adjunct Grammars. *J. Comput. Syst. Sci.* 10(1).

Kaplan, R. and Bresnan, J., 1983. Lexical-functional Grammar: A Formal System for Grammatical Representation. In Bresnan, J. (editor), *The Mental Representation of Grammatical Relations.* MIT Press, Cambridge MA.

Karttunen, Lauri, 1986. *Radicals Lexicalism.* Technical Report CSLI-86-68, CSLI, Stanford University. To also appear in *New Approaches to Phrase Structures*, University of Chicago Press, Baltin, M. and Kroch A., Chicago, 1988.

Kroch, A. and Joshi, A. K., 1985. *Linguistic Relevance of Tree Adjoining Grammars.* Technical Report MS-CIS-85-18, Department of Computer and Information Science, University of Pennsylvania.

Pollard, Carl and Sag, Ivan A., 1987. *Information-Based Syntax and Semantics. Vol 1: Fundamentals.* CSLI.

Schabes, Yves and Joshi, Aravind K., June 1988. An Earley-Type Parsing Algorithm for Tree Adjoining Grammars. In $26^{th}$ *Meeting of the Association for Computational Linguistics (ACL'88).* Buffalo.

Schabes, Yves and Joshi, Aravind K., August 1989. The Relevance of Lexicalization to Parsing. In *Proceedings of the International Workshop on Parsing Technologies.* Pittsburgh.

Schabes, Yves; Abeillé, Anne; and Joshi, Aravind K., August 1988. Parsing Strategies with 'Lexicalized' Grammars: Application to Tree Adjoining Grammars. In *Proceedings of the* $12^{th}$ *International Conference on Computational Linguistics (COLING'88).* Budapest.

Steedman, M. J., 1985. Dependency and Coordination in the Grammar of Dutch and English. *Language* 61:523–568.

Steedman, M., 1987. Combinatory Grammars and Parasitic Gaps. *Natural Language and Linguistic Theory* 5:403–439.

Vijay-Shanker, K., 1987. *A Study of Tree Adjoining Grammars.* PhD thesis, Department of Computer and Information Science, University of Pennsylvania.